

IN THE CLAIMS

Please amend the claims as follows:

1-2. (canceled)

3. (previously presented) The method of claim 14, wherein storing variations is performed using a pointer.

4. (currently amended) The method of claim 14, further comprising, before presenting the reconstructed represented sequence to a user:
identifying, replets that can be used to represent multiple subsequences.

5. (currently amended) The method of claim 14, further comprising, before presenting the reconstructed represented sequence to a user:
segmenting the matching subsequences into multiple parts to account for location-specific variations of the matching subsequences in the sequence data.

6. (currently amended) The method of claim 14, further comprising, before presenting the reconstructed represented sequence to a user:
storing replet information in a replet-information table using a pointer, so that equivalent replet sequences occupy single storage space.

7- 9. (canceled)

10. (currently amended) The method of claim 14, further comprising, before presenting the reconstructed represented sequence to a user:
storing multiple views of the sequence data at multiple levels of abstraction.

11-13. (canceled)

14. (currently amended) A computer system-implemented method for storing and presenting sequence data, comprising:

i) specifying a set of ~~one or more~~ initial-replets ~~for analysis by a computer system~~;

ii) for each initial-replet in the set, comparing ~~each~~ an initial-replet ~~by the computer system~~ to a sequence ~~to for determining by the computer system~~ a subsequence of the sequence that matches ~~each~~ the initial-replet, if any;

iii) generating ~~and storing~~ entries of match-set data structures ~~by the computer system responsive to the comparing~~, one match-set data structure for each replet, a match-set data structure comprising a sequence identification to identify a subsequence where a match occurs and offset information to determine a position within the sequence where the matching subsequence of the sequence is located, wherein the offset information comprises a first and second position parameter, the first position parameter denoting a location in the sequence and the second position parameter denoting an offset from the location;

~~, and wherein redundant match-set data structures corresponding to subsequences being matched by more than one replet are not generated;~~

iv) storing the generated entries of the match-set data structures in a computer readable memory by the computer system;

~~storing one or more variations for each of one or more matching subsequences, wherein a variation comprises a character in a subsequence that corresponds to a "don't care" character in a replet that matches the subsequence;~~

v) deleting by the computer system each matching subsequence from the sequence where it is found;

vi) concatenating by the computer system, in order, unmatched regions of sequences that remain after deleting each matching subsequence to form and store a backbone sequence;

vii) reconstructing the computer system representing the sequence ~~a sequence from responsive to the stored backbone sequence and at least a portion of the stored match-set data, the stored variations, and the stored backbone sequence~~; and presenting the reconstructed ~~represented~~ sequence to a user of the computer system;

wherein one of the subsequences is matched by a certain plurality of the replets and the method further includes:

viii) the computer system generating or receiving a selection of one of the certain plurality of replets, wherein any non-selected one of the certain plurality of replets is deemed a redundant replet and the representing in step vii) is responsive to the selected one of the certain plurality of replets but not responsive to any redundant replet;

ix) the computer system generating or receiving a selection of one of the replets deemed a redundant replet in step viii); and

x) the computer system representing and presenting the sequence, wherein the representing in step x) is responsive to the one of the replets indicated by the selection in step ix) instead of the one of the replets indicated by the selection in step viii) and wherein the computer system performs the representing in step x) by reference to the first and second parameters for the one of the replets indicated by the selection in step ix), such that the computer system performs the representing in step x) without repeating the generating of position parameters performed in step iii).

15. (currently amended) The method of claim 14, wherein, prior to presenting the reconstructed represented sequence to the user, variations are stored in a list data structure comprising a variation identification.

16. (previously presented) The method of claim 15, wherein the list data structure comprises a subsequence character that matches a "don't care" character in a replet that matches the subsequence.

17. (currently amended) The method of claim 16, wherein, prior to presenting the ~~reconstructed~~-represented sequence to the user, the position of the subsequence character within the subsequence is stored in the list data structure.

18. (previously presented) The method of claim 15, wherein an indirection pointer points to a variation so that variations common to more than one subsequence are not stored more than once.

19. (currently amended) The method of claim 14, wherein a ~~reconstructed~~-represented sequence presented to the user is in response to a query by the user.

20. (previously presented) The method of claim 19, wherein a query specifies a replet.

21. (currently amended) The method of claim 14, wherein specifying a set of one or more ~~initial~~-replets comprises adding a new replet to a pre-existing set for which the steps of claim 14 have been performed.

22. (currently amended) The method of claim 14, wherein specifying a set of one or more ~~initial~~-replets comprises removing a replet from a pre-existing set for which the steps of claim 14 have been performed.

23. (canceled)